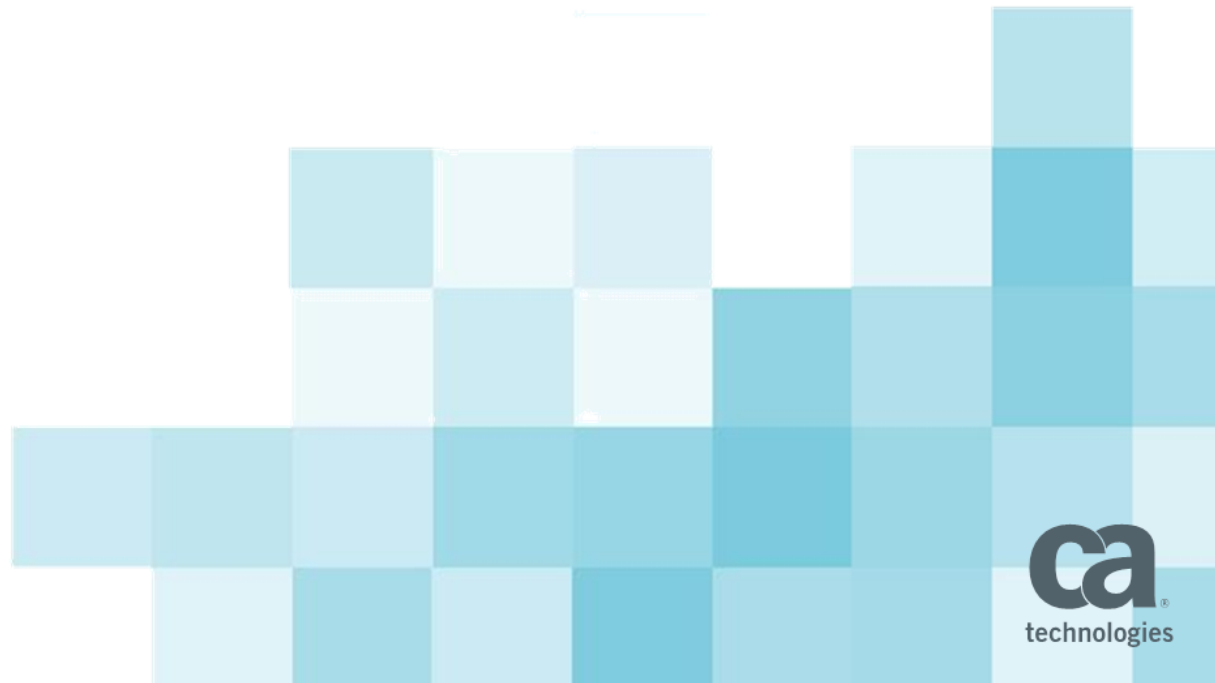


# CA IDMS™ 19.0

## Web Services for Modernization

Dave Ross

CA Technologies



# Abstract

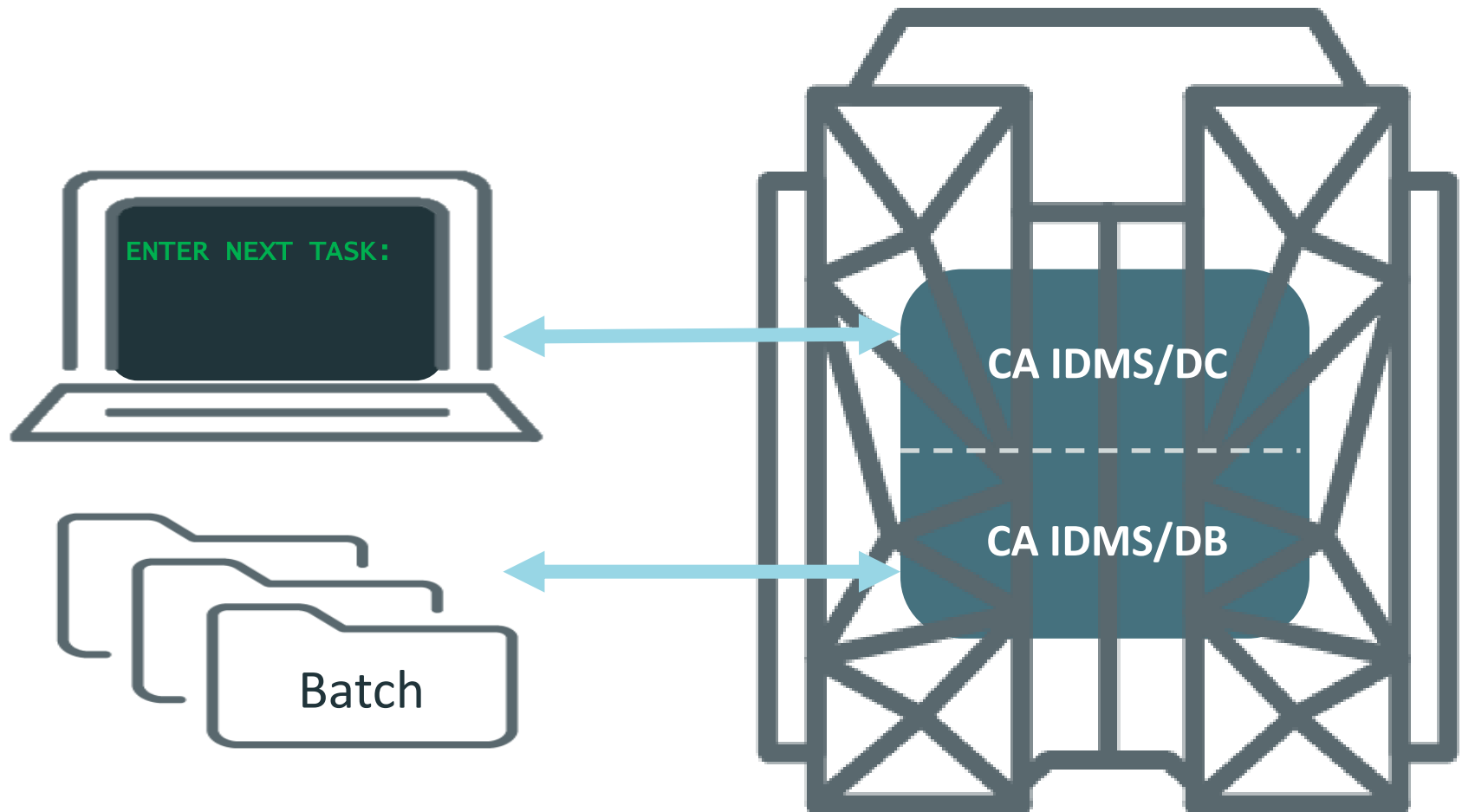
The goal for CA IDMS 19.0 is to improve CA IDMS modernization capabilities through features that enable customers to expand investments in core CA IDMS applications and improve developer productivity using modern skills and industry-standard technology.

# Agenda

- Leveraging Your Investment with Web Services
- CA IDMS Web Services
- XML Generation and Parsing
- Web Services Demo

# CA IDMS Applications

*Then...*



# Apps Everywhere



# CA IDMS Applications

*Now...*



# How Do You Maximize Business Value?

## Large investment in legacy applications

- Costly and risky conversion
- Hard to find legacy skills

## Leverage and Extend

- Leverage investment, reduce cost
- Preserve applications, reduce risk
- Use current developer skills

# Leveraging and Extending CA IDMS

## Leverage CA IDMS databases

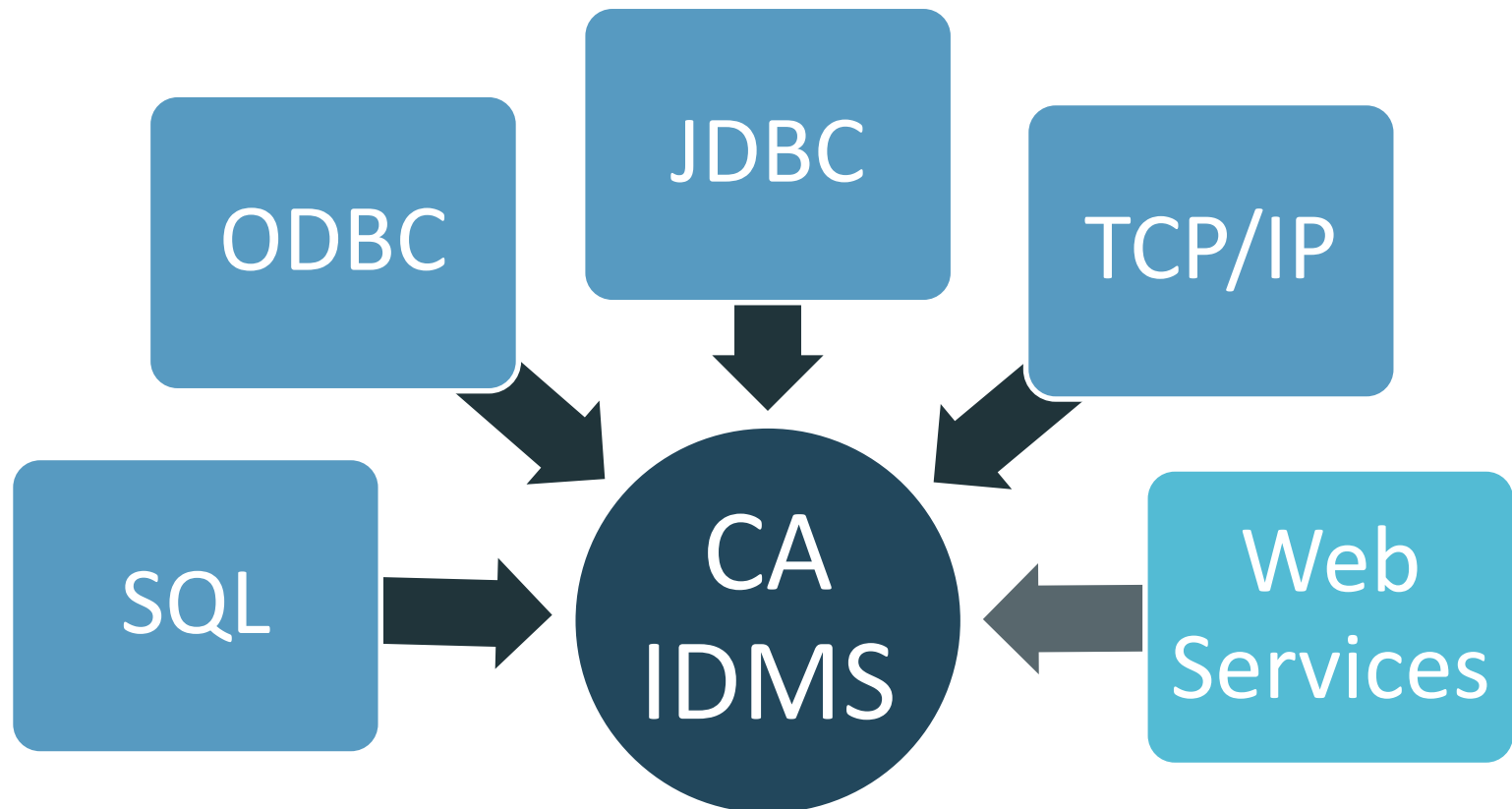
- Keep your database in place
  - Access from web services
  - Use standard interfaces

## Extend CA IDMS applications

- Reuse your application business logic
  - Invoke web services
  - Provide web services



# The CA IDMS Application Server



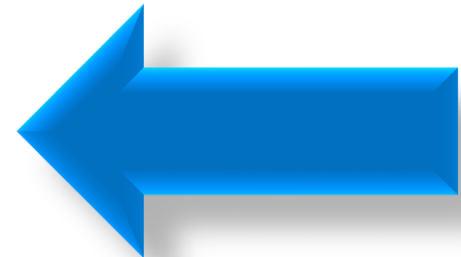
# What are Web Services?

- Support business processes
- Are invoked programmatically over the Internet
- Loosely coupled for interoperability
- Can be combined to build applications
- Implemented using SOAP, JSON, REST as well as other lower-level protocols



# Web Services participants

- The calling program
  - Consumer
  - Requester
  - Sometimes called outbound Web services
- The responding (or 'called') program
  - Provider
  - Producer
  - Sometimes called inbound Web services
  - Sometimes called the 'service implementation'



# Web Services Terminology

- SOAP: Simple Object Access Protocol
  - Uses both the XML and HTTP protocols
  - May include a WSDL file
- REST: Representational State Transfer
  - Lighter weight
  - Good for internal, loosely coupled and mobile services
  - May be stateless
- WSDL: Web Services Definition Language
- HTTP: Hyper Text Transfer Protocol
  - SOAP over HTTP – better for external Web services, not guaranteed
  - SOAP over JMS – better for internal use, guaranteed response
- XML: eXtensible Markup Language
  - Defines a method for encoding data in a program-readable, as well as human-readable format.

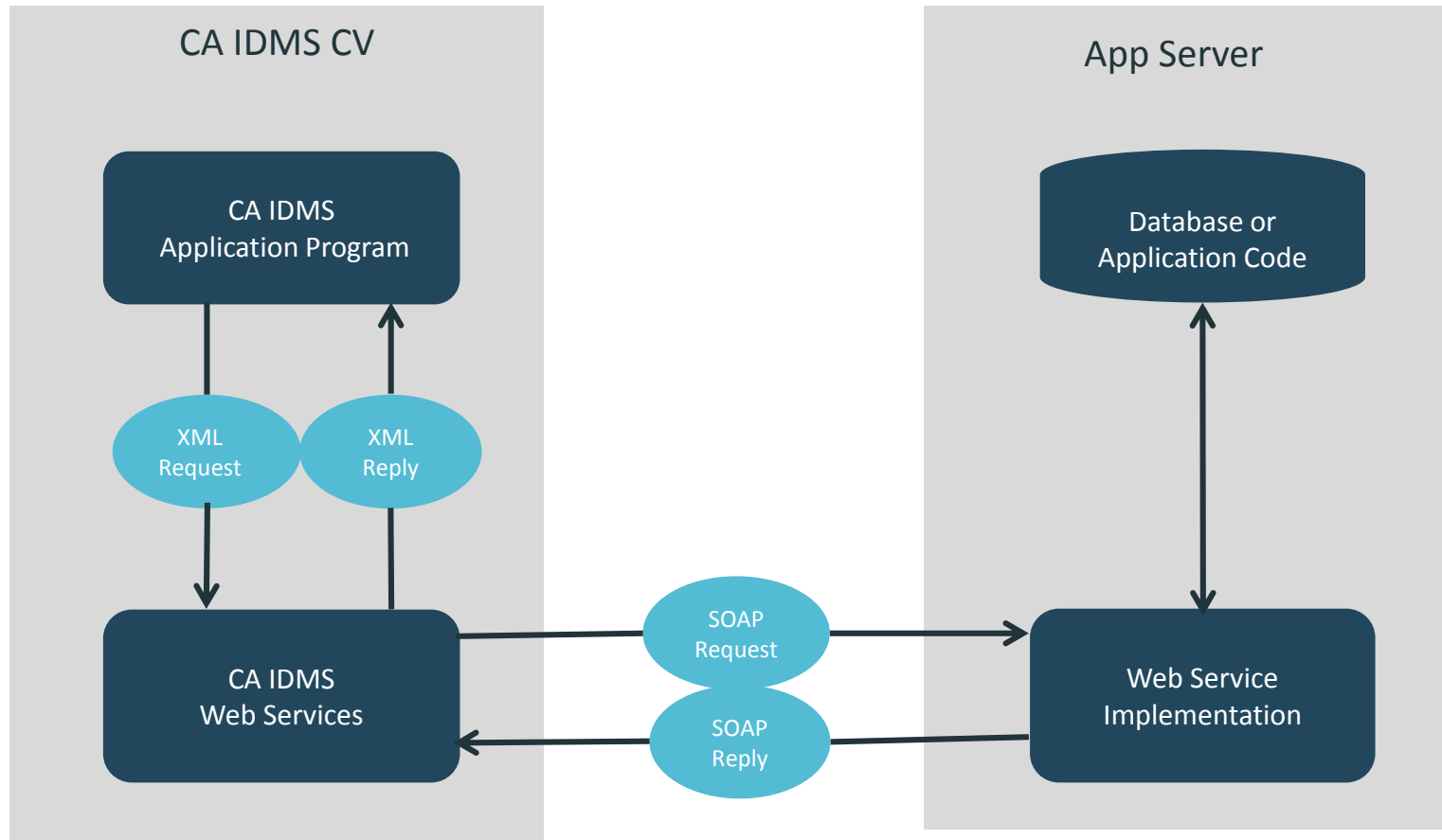


# CA IDMS Web Services

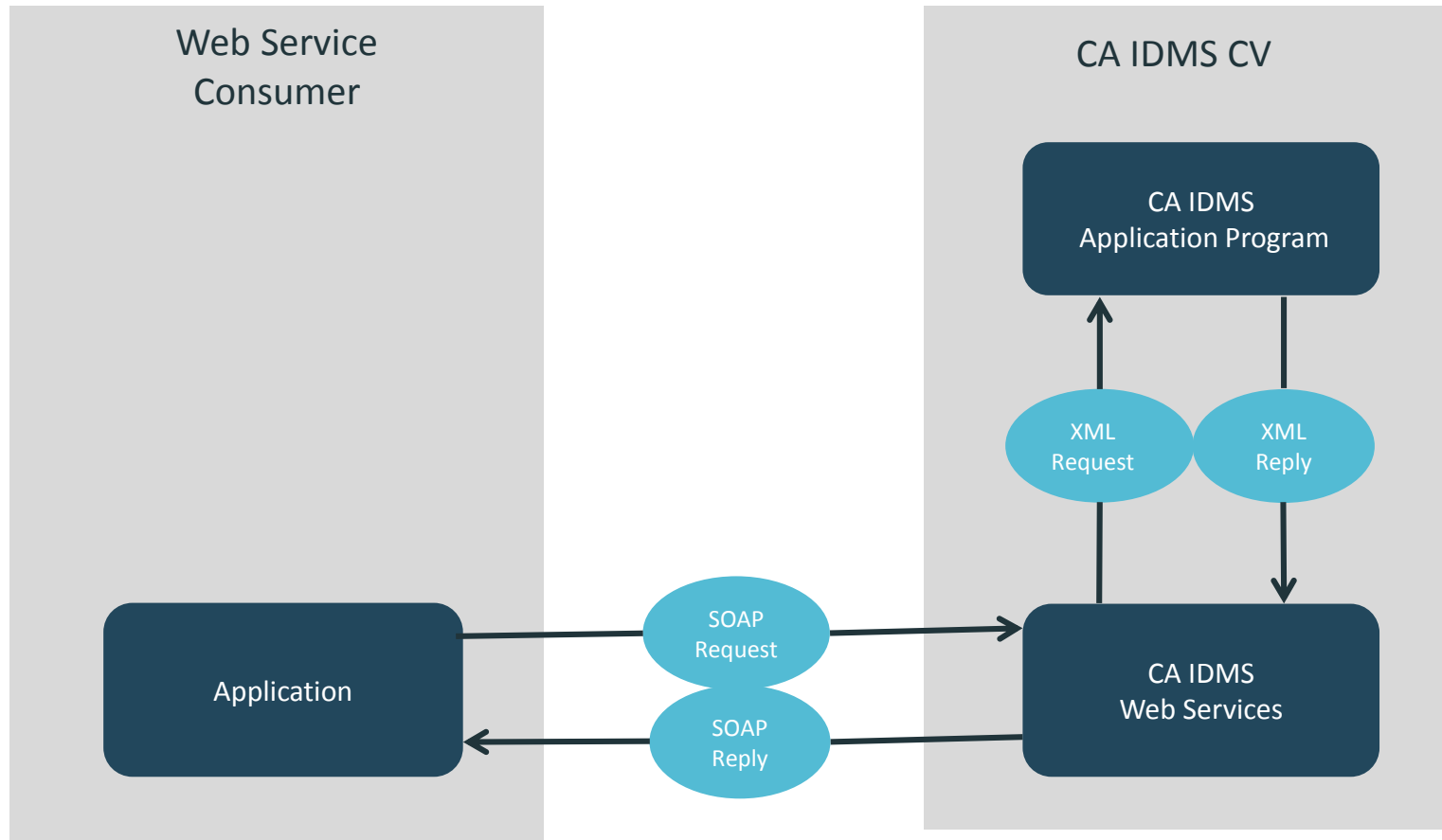
# CA IDMS Web Services

- Web Service Consumer
- Web Service Provider
- Web Services API
- Leverage and extend CA IDMS applications
  - COBOL
  - ADS
  - PL/1
  - Assembler

# CA IDMS as a Web Service Consumer



# CA IDMS as a Web Service Provider





# The CA IDMS Web Services API



# Web Services API

- Well defined, extendable interface
  - Simplifies application development
  - Isolate user code from product changes
  - Provide consistent base for product enhancement
- CA IDMS Callable Service
  - COBOL, ADS, PL/1, Assembler
- WS API Functions
  - Data transfer
  - Session management
  - Option management

# Web Services API Functions

Function Code	Description	Used By
4	INITIALIZE	Consumer/Provider
8	SETOPTION	Consumer/Provider
12	GETOPTION	Consumer/Provider
16	REQUEST	Consumer
20	SEND	Provider
24	RECEIVE	Provider
28	RELEASE	Consumer/Provider

# Using the Web Services API

Consumer		Provider	
Operation	API Function	Operation	API Function
Initialize Environment	WSINITIALIZE	Initialize Environment	WSINITIALIZE
Manage Options	WSGETOPTION WSSETOPTION	Manage Options	WSGETOPTION WSSETOPTION
Send Request	WSREQUEST	Receive Request	WSRECEIVE
and Receive Response		Send XML Response	WSEND
Free Resources	WSRELEASE	Free Resources	WSRELEASE

# Invoking the Web Services API

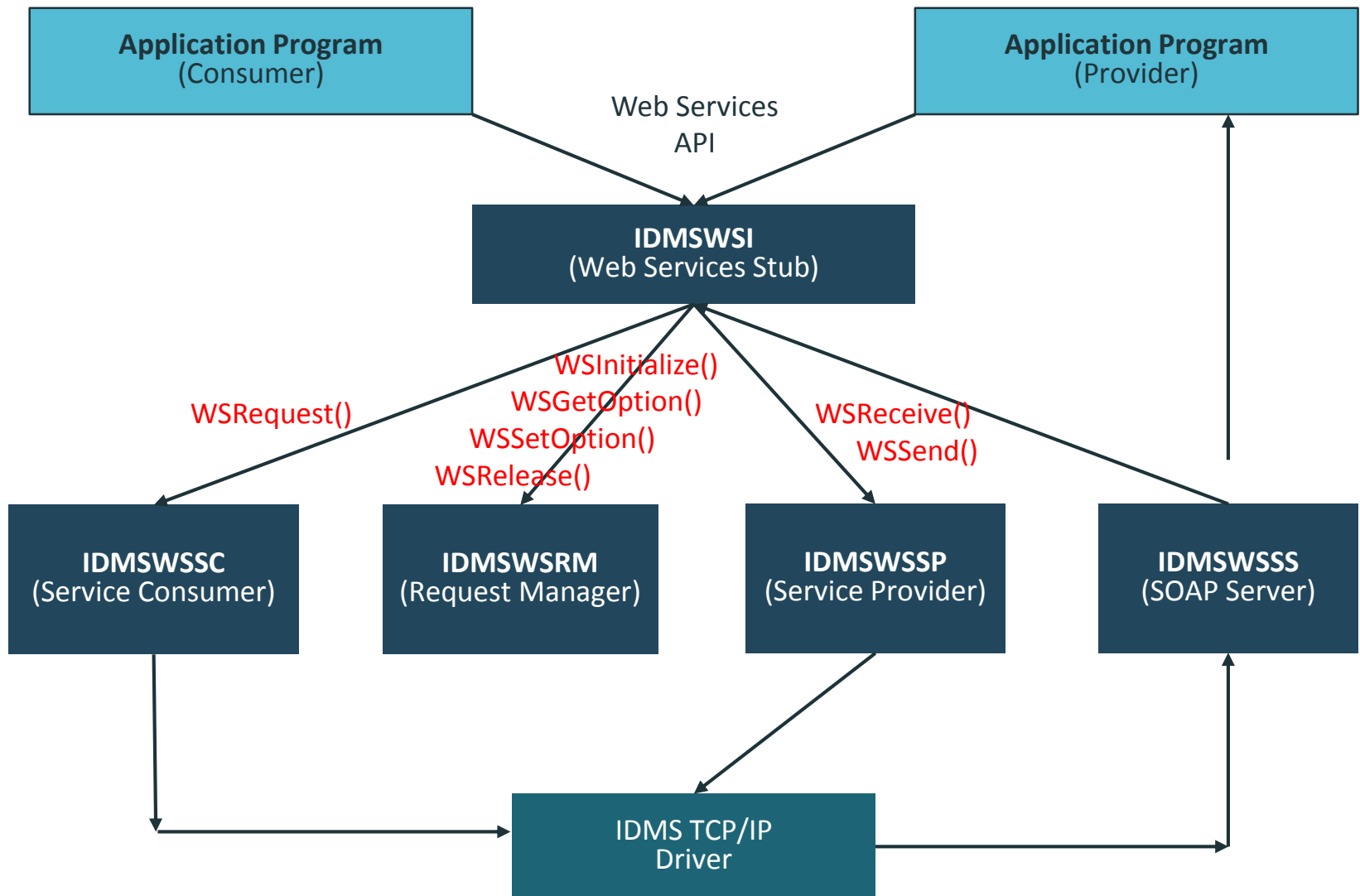
## COBOL

```
CALL 'IDMSWSI' USING  
    function,  
    return-code,  
    error-info,  
    function-dependent-parameter1,  
    . . .
```

## ADS

```
LINK TO PROGRAM 'IDMSWSI' USING  
    function,  
    return-code,  
    error-info,  
    function-dependent-parameter1,  
    . . .
```

# CA IDMS Web Services Internal Architecture



# Web Services API Records

- Every API call will have the following 4 records at the beginning of each call and a variation of the additional records following.

•WS-FUNCTION-CODE-RECORD	API Function Code
•WS-RETURN-CODE-RECORD	API Return Code
•WS-ERROR-INFO	API Error Information
•WS-INTERFACE-VERSION-NUMBER	API Interface version
•WS-REQUEST-INFO	Request SOAP information
•WS-REQUEST-HANDLE-RECORD	Request Handle record
•WS-OPTION-NUMBER-RECORD	Get/Set Option number
•WS-OPTION-VALUE-RECORD	Get/Set Option value
•WS-REQUEST-MSG-DATA (Module)	Request Message
•WS-REQUEST-MSG-PTR-RECORD	Request Pointer
•WS-RESPONSE-MSG-DATA (Module)	Response Message
•WS-RESPONSE-MSG-PTR-RECORD	Response Pointer
•WS-REQUEST-MSG-DESCRIPTOR	Request Length
•WS-RESPONSE-MSG-DESCRIPTOR	Response Length

# Web Services API Return Codes

Return Code Value	Severity	Description
0	Successful	Successful return
4	Warning	Request processed, warning msg issued
8	Error	Request fails, error message returned
12	Critical	Request fails, Service terminated
16	Systemic	Request fails, impact to all Services



# Web Services API Error Information

- WS-ERROR-INFO provides additional fields that define the result of the call

## Error Type:

INTERNAL (I) Generated from failures in CA IDMS/DC operations

API (A) Failure to adhere to Web Services API protocol

XML (X) Generated if XML Parsing or Generation fails

HTTP (H) API receives an unexpected HTTP status code

TCPIP (T) An unexpected TCPIP code received

SOAP (S) An unexpected SOAP fault code received

OTHER (O) An unclassified error occurred

**Error Text:** Text that describes additional content to the error

# Web Services API – INITIALIZE (4)

- Allocate and initialize Web Services data structures

## **Example for COBOL, ADS and PL/I**

WS-INITIALIZE,  
return-code,  
error-info,  
request-handle,  
interface-version.

# Web Services API – SETOPTION (8)

- Dynamically override default settings of the CA IDMS Web services system-level options

Option Name	Number	Description
LOG-SERVICES	1	Turn Web Services Logging on or off
LOG-PROGRAM	2	Log Specific program
REQUIRE-SIGNON	3	Require CV logon
CHECK-AUTH	4	Requires that User is part of Services security Group
CONNECT-TIMEOUT	5	Specify wait time for external services
READ-WRITE-TIMEOUT	6	Specify wait time for TCP/IP calls
XML-CODE-PAGE	7	Set codepage value for XML Processing

**Example: COBOL, ADS and PL/I**  
WS-SETOPTION,  
return-code,  
error-info,  
request-handle,  
option-number,  
option-value.

# Web Services API – GETOPTION (12)

- GETOPTION retrieves the values for the Web Services system-level options.

## **Example for COBOL, ADS and PL/I**

WS-GETOPTION,  
return-code,  
error-info,  
request-handle,  
option-number,  
option-value.

# Web Services API – REQUEST (16)

- The REQUEST function builds and transmits a SOAP service request.

## **Example: COBOL, ADS and PL/I**

WS-REQUEST,  
return-code,  
error-info,  
request-handle,  
request-info,  
request-message-data,  
request-message-descriptor,  
response-message-data,  
response-message-descriptor.

# Web Services API – SEND (20)

- The SEND function is used to transmit a Response message to a service Consumer

## **Example: COBOL, ADS and PL/I**

WS-SEND,  
return-code,  
error-info,  
request-handle,  
response-message-data,  
response-message-descriptor.

# Web Services API – RECEIVE (24)

- The RECEIVE function is used to return the address and length of an incoming Web service Request buffer

## **Example: COBOL, ADS and PL/I**

WS-RECEIVE,

return-code,

error-info,

request-handle,

request-message-data,

request-message-descriptor.

# Web Services API – RELEASE (28)

- The RELEASE function is used to terminate a Web Services request. It frees all structures allocated on behalf of the Web Services request

## **Example: COBOL, ADS and PL/I**

WS-RELEASE,  
return-code,  
error-info,  
request-handle.



# XML Generation and Parsing

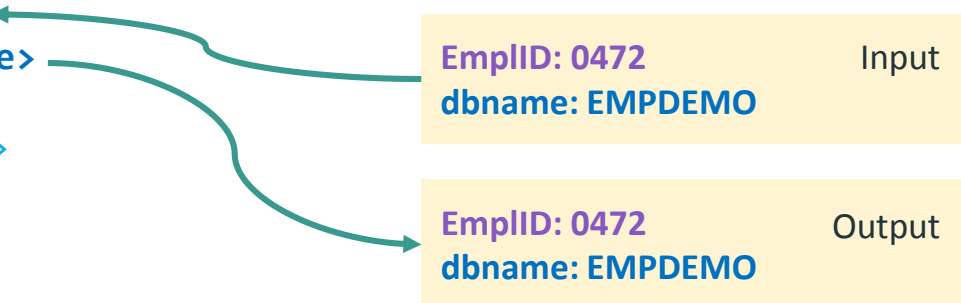
# XML Message Contents

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  <soap:Body>
    <IDMSWSPIOperation
      xmlns="http://www.IDMSWSPI.Request.com">
        <InputFields>
          <EmplID>0472</EmplID>
          <dbname>EMPDEMO</dbname>
        </InputFields>
      </IDMSWSPIOperation>
    </soap:Body>
  </soap:Envelope>
```

# XML Generation and Parsing

- XML Generation
  - Uses input variables to create an XML Message
  - Final message includes your data as the payload

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body><IDMSWSPIOperation xmlns="http://www.IDMSWSPI.Request.com">
    <InputFields>
      <EmplID>0472</EmplID>
      <dbname>EMPDEMO</dbname>
    </InputFields>
  </IDMSWSPIOperation></soap:Body>
</soap:Envelope>
```



- XML Parsing
  - Extracts the payload from a given XML message
  - Payload is interpreted as individual variables for use Web services
- Two main approaches: use of IBM's COBOL functions or use of built-in SQL/XML functions

# Use of SQL/XML Functions for XML Generation

0000-MAINLINE-BEG.

EXEC SQL

set :DOC=

xmlserialize(content

xmlelement(name "soapenv:Envelope"

, xmlnamespaces(

'http://schemas.xmlsoap.org/soap/envelope/'

as "soapenv"

, default 'HTTP://CA.COM/HR/GLOBALXML'),

, xmlelement(name "soapenv:Header")

, xmlelement(name "soapenv:Body"

, xmlelement(name "IDMSWSPIOperation"

, xmlnamespaces('http://www.IDMSWSPI.Request.com'

, xmlconcat(xmlelement(name "InputFields"

, xmlconcat(xmlelement(name "EmplID", :EMPLID)

xmlelement(name "dbname", :DBNAME)

)))))) as char(1000))

END-EXEC.

COBOL paragraph name

01 DOC PIC X(1000) VALUE SPACES.

Example of generation of an XML message

01 EMPLID PIC X(4).  
01 DBNAME PIC X(8).

# XML GENERATE

- Generates XML document from COBOL data structure
- Use to return CA IDMS data in XML format
- XML GENERATE introduced in COBOL for z/OS V3R3
  - Also supported for PL/1
- Compile with XMLSS to use z/OS XML system services parser

# COBOL XML Generate Example

Using an example from the CA Web Services Demo Provider program ...

1. A COBOL data structure is defined containing the fields extracted from the Employee Demo Data base required for the Provider Service. CA Web Services provides routines that will execute the XML Generate for the COBOL data structure '*OutputFields*'. Just define the output data under this 01 level.

```
01  OutputFields.  
   05  EmpID          PIC X(04)  VALUE SPACES.  
   05  EmpFirstName   PIC X(10)  VALUE SPACES.  
   05  EmpLastName    PIC X(15)  VALUE SPACES.  
   05  EmpStreet      PIC X(20)  VALUE SPACES.  
   05  EmpCity        PIC X(15)  VALUE SPACES.  
   05  EmpState       PIC X(02)  VALUE SPACES.  
   05  EmpZip         PIC X(05)  VALUE SPACES.
```

2. The data structure fields are case sensitive, the above fields will appear with upper/lower case in the XML data tag.  
<EmpFirstName> </EmpFirstName>

# COBOL XML Generate

Using an example from the CA Web Services Demo Provider program ...

1. The COBOL XML Generate Statement creates an XML Response from the COBOL data structure.

```
XML GENERATE WS-RESPONSE-MESSAGE
  (1:WS-RSP-MSG-BUFF-LEN)
  FROM OutputFields          Defined COBOL data structure
  COUNT IN WSPI-XML-OUT-LENGTH
  WITH ENCODING WS-CODEPAGE-VALUE
  ON EXCEPTION
    MOVE 'NO ' TO WSPI-WAS-GENERATE-SUCCESS
```

2. The resulting XML structure is stored in WS-RESPONSE-MESSAGE where it can be wrapped by a SOAP Envelope as a Service Response

```
<OutputFields><EmpID>0472</EmpID>
><EmpFirstName>ROBBY</EmpFirstName><EmpLastName>WILDER
</EmpLastName><EmpStreet>4567 E. GROWTH ST</EmpStreet>
<EmpCity>SOUTHBORO</EmpCity><EmpState>MA</EmpState><EmpZip>03145</EmpZip></OutputFields>
```

# COBOL XML Parse

COBOL XML PARSE transforms XML String into COBOL data items.

**XML string**

**<Emp | ID>0472</Emp | ID**

**COBOL PARSE XML string**

```
XML PARSE CLA1-REPLY-BUFFER
  (1:CWA1-REPLY-BUFFER-LENGTH)
  WITH ENCODING CWA1-CODEPAGE-VALUE
  PROCESSING PROCEDURE CPA1-PARSE-XML
  ON EXCEPTION
    MOVE 'NO ' TO CWA1-WAS-PARSE-SUCCESSFUL
```

**Evaluate the data tags and populate data into COBOL data structure**

```
EVALUATE FUNCTION UPPER-CASE(CWA1-EDITED-ELEMENT)
  WHEN 'EMPLID'
    MOVE XML-TEXT TO WS-EMP-ID
```

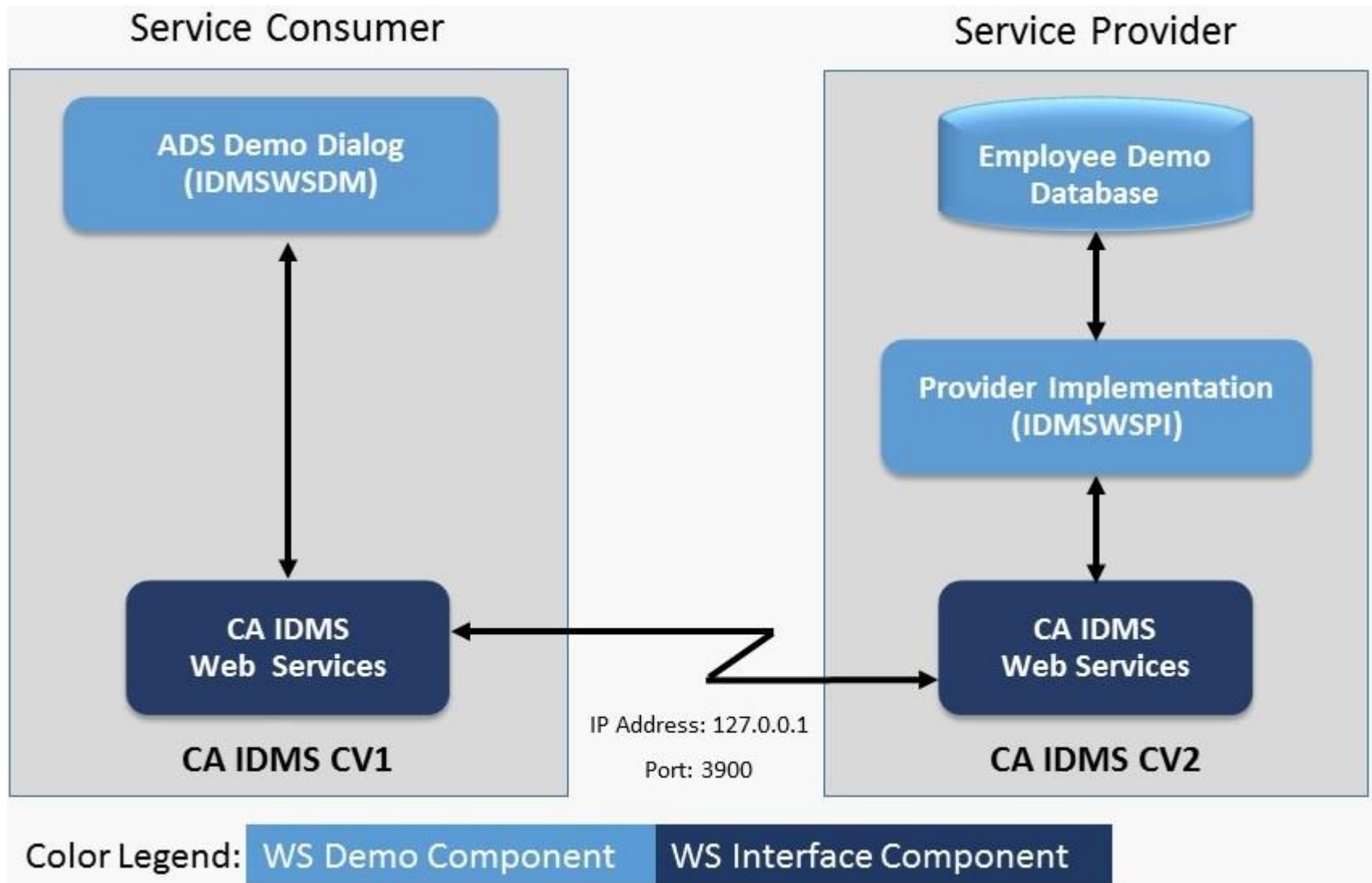
**Results in:**

**WS-EMP-ID = 0472**



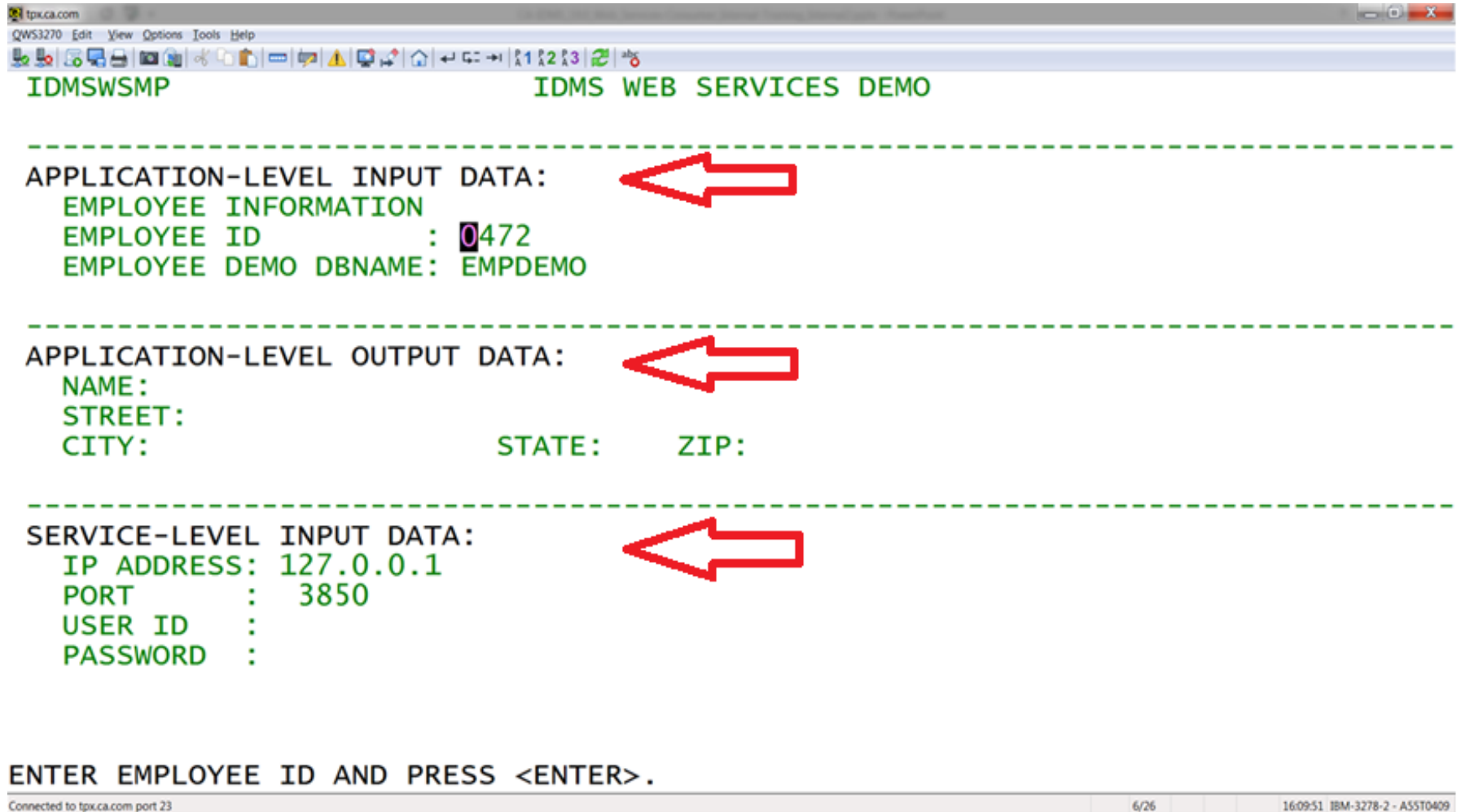
# Web Services Demo Programs

# CA IDMS Web Services Demo



# CA IDMS Web Services Demo

- Enter task code 'IDMSWSDM'




tpx.ca.com

QWS3270 Edit View Options Tools Help

IDMSWSMP IDMS WEB SERVICES DEMO

-----


APPLICATION-LEVEL INPUT DATA: 

EMPLOYEE INFORMATION

EMPLOYEE ID : 0472

EMPLOYEE DEMO DBNAME: EMPDEMO

-----


APPLICATION-LEVEL OUTPUT DATA: 

NAME:

STREET:

CITY: STATE: ZIP:

-----

SERVICE-LEVEL INPUT DATA: 

IP ADDRESS: 127.0.0.1

PORT : 3850

USER ID :

PASSWORD :

ENTER EMPLOYEE ID AND PRESS <ENTER>.

Connected to tpx.ca.com port 23 6/26 16:09:51 IBM-3278-2 - ASST0409

# CA IDMS Web Services Demo

- CA IDMS Web Services Consumer receives reply from CA Web Services Provider Service.



## APPLICATION-LEVEL INPUT DATA:

EMPLOYEE INFORMATION

EMPLOYEE ID : 0472

EMPLOYEE DEMO DBNAME: EMPDEMO

## APPLICATION-LEVEL OUTPUT DATA:

NAME: ROBBY WILDER

STREET: 4567 E. GROWTH ST

CITY: SOUTHBORO STATE: MA ZIP: 03145

## SERVICE-LEVEL INPUT DATA:

IP ADDRESS: 127.0.0.1

PORT : 3850

USER ID :

PASSWORD :

ENTER ANOTHER EMP ID AND PRESS <ENTER>.

Connected to tpx.ca.com port 23

6/26

16:11:04 IBM-3278-2 - A55T0409

# CA IDMS Web Services Demo – Request API

- To send a Service Request for the Consumer, the ADS dialog uses the Web Service Request API

```
!*  WSREQUEST()          - PERFORM A REQUEST TO CONSUME A WEB SERVICE
!*****
MOVE 16 TO WS-FUNCTION-CODE.          ! WSREQUEST()
LINK TO PROGRAM 'IDMSWSI' USING
    (WS-FUNCTION-CODE-RECORD,
     WS-RETURN-CODE-RECORD,
     WS-ERROR-INFO,
     WS-REQUEST-HANDLE-RECORD,
     WS-REQUEST-INFO,
     WSDemo-REQUEST-MSG-DATA,
     WS-REQUEST-MSG-DESCRIPTOR,
     WSDemo-RESPONSE-MSG-DATA,
     WS-RESPONSE-MSG-DESCRIPTOR) .
```

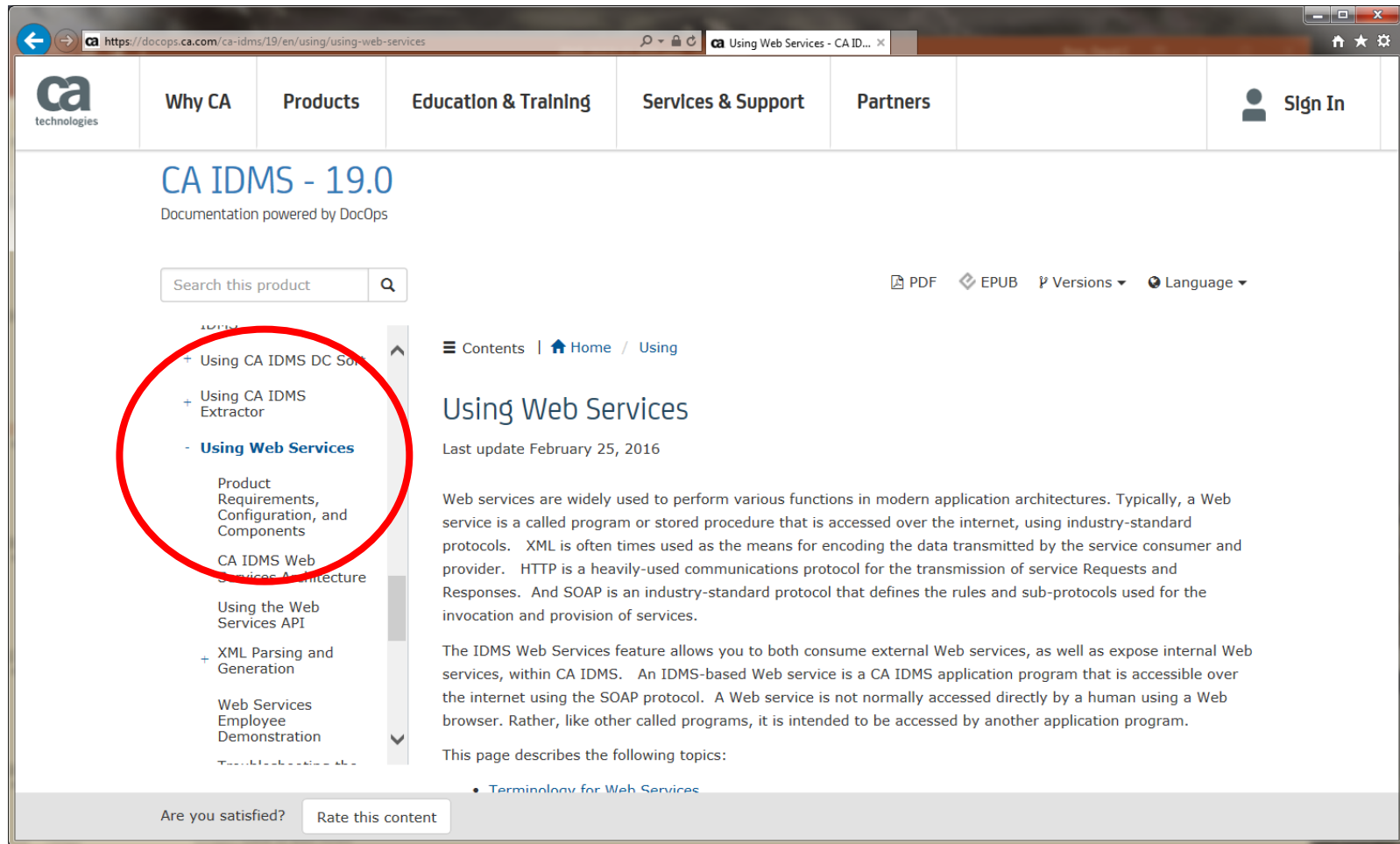
**Defines Service Request SOAP information**  
**Defines Service Request message**  
**Defines Service Request Length**  
**Defines Service Response message**  
**Defines Service Response Length**

# Web Services Documentation

https://docops.ca.com/idms

The screenshot shows a web browser window displaying the CA IDMS 19.0 documentation page. The browser's address bar shows the URL <https://docops.ca.com/ca-idms/19/en>. The page features a navigation bar with links: Why CA, Products, Education & Training, Services & Support, Partners, and a Sign In button. Below the navigation bar is a dark blue header with the text "CA IDMS - 19.0" and "Documentation powered by DocOps". A search bar is located below the header, with the text "Search this product" and a magnifying glass icon. To the right of the search bar are links for "Versions" and "Language". On the left side, there is a vertical menu with a scroll bar, containing links: CA IDMS Guides to DocOps Mapping, + Release Notes, Post Release Updates, + Product Information, + Installing, + Using, + Administrating, + Reporting, Messages, + Additional Resources, Reference, Documentation Legal Notice, and a dropdown arrow. The main content area has a "Contents | Home" link. Below this is a section titled "Announcements & News" with a list of items: "Beginning with CA IDMS 19.0, Documentation is Online", "PDFs and EPUBs are Back!", and "Guides to DocOps Mapping". Below this are two sections: "Release Notes" and "Installing". The "Release Notes" section contains the text: "New and enhanced features. Install and upgrade considerations. Documentation changes." The "Installing" section contains the text: "Install, configure, and maintain IDMS server and components using [best practices](#)." At the bottom of the page, there is a footer with a "Rate this content" button and a "Are you satisfied?" link. The footer also includes a "Using" link and an "Administrating" link.

# Using Web Services



The screenshot shows the CA IDMS 19.0 documentation website. The browser address bar displays <https://docops.ca.com/ca-idms/19/en/using/using-web-services>. The navigation bar includes links for Why CA, Products, Education & Training, Services & Support, Partners, and a Sign In button. The main heading is "CA IDMS - 19.0" with the subtext "Documentation powered by DocOps". A search bar is present with the text "Search this product". On the right, there are links for PDF, EPUB, Versions, and Language. The left sidebar contains a list of topics, with "Using Web Services" highlighted by a red circle. The main content area is titled "Using Web Services" and includes a "Last update February 25, 2016" note. The text describes web services as programs accessed over the internet using standard protocols like XML, HTTP, and SOAP. It also mentions the IDMS Web Services feature for consuming and exposing services. At the bottom, there is a "Rate this content" button and a link to "Terminology for Web Services".

CA technologies

Why CA Products Education & Training Services & Support Partners Sign In

CA IDMS - 19.0  
Documentation powered by DocOps

Search this product

PDF EPUB Versions Language

Contents | Home / Using

Using Web Services

Last update February 25, 2016

Web services are widely used to perform various functions in modern application architectures. Typically, a Web service is a called program or stored procedure that is accessed over the internet, using industry-standard protocols. XML is often times used as the means for encoding the data transmitted by the service consumer and provider. HTTP is a heavily-used communications protocol for the transmission of service Requests and Responses. And SOAP is an industry-standard protocol that defines the rules and sub-protocols used for the invocation and provision of services.

The IDMS Web Services feature allows you to both consume external Web services, as well as expose internal Web services, within CA IDMS. An IDMS-based Web service is a CA IDMS application program that is accessible over the internet using the SOAP protocol. A Web service is not normally accessed directly by a human using a Web browser. Rather, like other called programs, it is intended to be accessed by another application program.

This page describes the following topics:

- Terminology for Web Services

Are you satisfied? Rate this content



# Web Services API Reference

CA technologies

Why CA Products Education & Training Services & Support Partners Sign In

## CA IDMS Reference - 19.0

Documentation powered by DocOps

Search this product

PDF EPUB Versions Language

Contents | Home / Callable Services Reference

### Web Services API Support

Last update February 16, 2016

The CA IDMS Web Services API is a call-level interface for the programmatic invocation and provision of Web services within CA IDMS. This section includes the following topics:

- [Web Services API Functions](#)
- [Web Services API Records](#)
- [Supported Programming Languages](#)

The following programming languages, supported within CA IDMS, can use the IDMS Web Services API:

- Assembler
- CA IDMS ADS/Online

Are you satisfied? Rate this content

# Summary

- Leveraging Your Investment with Web Services
- CA IDMS Web Services
- XML Generation and Parsing
- Web Services Demo

# FOR INFORMATION PURPOSES ONLY

## Terms of this Presentation

This presentation was based on current information and resource allocations as of September 2016 and is subject to change or withdrawal by CA at any time without notice. Notwithstanding anything in this presentation to the contrary, this presentation shall not serve to (i) affect the rights and/or obligations of CA or its licensees under any existing or future written license agreement or services agreement relating to any CA software product; or (ii) amend any product documentation or specifications for any CA software product. The development, release and timing of any features or functionality described in this presentation remain at CA's sole discretion. Notwithstanding anything in this presentation to the contrary, upon the general availability of any future CA product release referenced in this presentation, CA will make such release available (i) for sale to new licensees of such product; and (ii) to existing licensees of such product on a when and if-available basis as part of CA maintenance and support, and in the form of a regularly scheduled major product release. Such releases may be made available to current licensees of such product who are current subscribers to CA maintenance and support on a when and if-available basis. In the event of a conflict between the terms of this paragraph and any other information contained in this presentation, the terms of this paragraph shall govern.

Certain information in this presentation may outline CA's general product direction. All information in this presentation is for your informational purposes only and may not be incorporated into any contract. CA assumes no responsibility for the accuracy or completeness of the information. To the extent permitted by applicable law, CA provides this presentation "as is" without warranty of any kind, including without limitation, any implied warranties or merchantability, fitness for a particular purpose, or non-infringement. In no event will CA be liable for any loss or damage, direct or indirect, from the use of this document, including, without limitation, lost profits, lost investment, business interruption, goodwill, or lost data, even if CA is expressly advised in advance of the possibility of such damages. CA confidential and proprietary. No unauthorized copying or distribution permitted.

# Questions and Answers